Cuaderno de ejercicios: Optimización de código

Solución: Cuaderno de ejercicios: Optimización codigo

Ejercicio 1: Varios métodos a optimizar:

Optimiza los siguientes métodos.
Dificultad: ★★★

```
public class EjercicioOptimizacion {

   void ejemplo1SinOptimizar() {
       int b = 0, c = 0, d = 0;
       int a = b + c;
       d = a - d;
       int e = (b + c) * d;
   }

   void ejemplo2SinOptimizar() {
       int valor = 0, item;
       do {
           item = 10;
           valor += valor + item;
       } while (valor < 100);
   }

   void ejemplo3SinOptimizar() {
       String total = "";
       for (int i = 0; i < 10; i++) {
           // otras operaciones....
```

```
            String m = "Mensaje hola";
            int contador = i;
            total = m + contador;
        }
        System.out.println(total);
    }

    void ejemplo4SinOptimizar() {
        int i = 2 + 3;
        int j = 4;
        float f = j + 2.5f;
        /* aquí hay más código que usa i,j,f */
    }

    void ejemplo5SinOptimizar() {
        int i = 10, c = 10, m = 10;
        int a = 3 + i;
        int f = a;
        int b = f + c;
        int d = a + m;
        m = f + d;
    }

    void ejemplo7SinOptimizar() {
        int i = 1;
        float array[] = new float[5];
        array[i] = array[8 + i] + (i + 1) * 5 * 8 + (5 + 1);
        array[i - 1] = array[8 + i] + (i + 1) * 5 * 8 + (6 + 1);
    }

    void ejemplo8SinOptimizar() {
        int pos = 3;
        int contador = 0;
        int array[] = new int[5];
        for (int i = 0; i < 2000; i++) {
            // cosas
            contador = array[pos] + i;
            // cosas
        }
        System.out.println(contador);
        // cosas
    }

    void ejemplo9SinOptimizar() {
        int a = 10;
        float b = a / 2;
    }

    int ejemplo10SinOptimizar() {
        int x = 10, y = 20;
        int z = x / y;
        return x / y;
    }

    void ejemplo11SinOptimizar() {
        int i = 0;
```

```
        while (i < 10) {
            System.out.println("hola:" + i);
            i++;
        }
        int j = 0;
        while (j < 10) {
            System.out.println("adios:" + j);
            j++;
        }
    }
}
```

Ejercicio 2: Trabajo con array de enteros:

Algunos de los métodos de este ejercicio tienen mejoras de optimización. Tu tarea será encontrarlas y solucionarlas.
Además añade un comentario (encima del método) indicando, mediante una breve reseña, que has hecho para solucionar el problema.
Dificultad: ★★★

```
import java.util.Arrays;

public class Optimiza {
    int numeros[] = { -5, 3, 6, 66, 55, 2, -7, 6, 1 };

    /* Este método busca si un numero está en una lista de números */
    boolean busca(int numeroBuscado) {
        boolean esta = false;
        for (int n : numeros) {
            if (numeroBuscado == n) {
                esta = true;
            }
        }
        return esta;
    }

    /* Este cuenta cuantos números positivos hay en la lista de numeros */
    int cuentaPositivos() {
        int contador = 0;
        int num = contador;
        for (int n : numeros) {
            if (n >= 0) {
                num = contador;
                contador++;
            }
        }
```

```
        num = num / numeros.length;
        return contador;
    }

    /*
     * Este método calcula la media de todos los números guardado en la lista de
     * números
     */
    float calculaMedia() {
        float cont = 0;
        for (double num : numeros) {
            cont += num;
        }
        return cont / numeros.length;
    }

    /*
     * Este método divide cada número de la lista entre la media de todos los
     * numeros
     */
    float[] dividelosPorLaMedia() {
        float nuevosNumeros[] = new float[numeros.length];
        for (int i = 0; i < numeros.length; i++) {
            nuevosNumeros[i] = numeros[i] / calculaMedia();
        }
        return nuevosNumeros;
    }

    /*
     * Este método calcula la mediana de la lista de numeros.
     * Recueda que la mediana representa el valor de la variable de posición central
     * en un conjunto de datos
     */
    double calculaMediana() {
        int[] copiedArray = numeros.clone();
        Arrays.sort(copiedArray);
        int mediana;
        if (copiedArray.length % 2 == 0) { // Si la longitud es par, se deben promediar
los del centro
            mediana = (copiedArray[copiedArray.length / 2 - 1] +
copiedArray[copiedArray.length / 2]) / 2;
        } else {
            mediana = copiedArray[copiedArray.length / 2];
        }
        return mediana;
    }

    int moda() {
        int maximaVecesQueSeRepite = 0;
        int moda = 0;
        for (int i = 0; i < numeros.length; i++) {
            int vecesQueSeRepite = 0;
            for (int j = 0; j < numeros.length; j++) {
                if (numeros[i] == numeros[j])
                    vecesQueSeRepite++;
            }
```

```
            if (vecesQueSeRepite > maximaVecesQueSeRepite) {
                moda = numeros[i];
                maximaVecesQueSeRepite = vecesQueSeRepite;
            }
        }
        return moda;
    }

    public static void main(String[] args) {
        new Optimiza();
    }

    public Optimiza() {
        System.out.println("Numeros: " + Arrays.toString(numeros));
        System.out.println("Tiene el 5:" + busca(5));
        System.out.println("Tiene el 2:" + busca(2));
        System.out.println("Hay " + cuentaPositivos() + " números positivos");
        System.out.println("La media vale: " + calculaMedia());
        System.out.println("Cada número dividido por la media de todos:" +
Arrays.toString(dividelosPorLaMedia()));

        System.out.println("La mediana vale:" + calculaMediana());
        System.out.println("La moda vale:" + moda());
    }
}
```

Ejercicio 3: Algoritmo:

Optimiza el siguiente método.
Dificultad: ★★

```
public class Optimizacion {
    public int[] algoritmo(int[] datos) {
        // float array[] = new float[10];
        int i = 1;
        for (i = 1; i < datos.length - 2; i++) {
            int sumador = 4 / 2;
            datos[i] = datos[2 + i] + (i + 1) * 2 * 8 + (5 + 2) + sumador;
            datos[i - 1] = datos[2 + i] + (i + 1) * 2 * 8 + (6 + 2) + sumador;
            int z = datos[2 + i] / sumador;
        }
        return datos;
    }

    public static void main(String[] arg) {
        Optimizacion ejemplo = new Optimizacion();
        int[] datos = new int[] { 1, 2, 3, 4, 5, 6, 100, 90 };
        ejemplo.algoritmo(datos);
```

```
        for (int d : datos) {
            System.out.println(d);
        }
    }
}
```

Ejercicio 4: Trasladar punto

Realiza primero los Test unitarios JUnit y luego optimiza el código.
Dificultad: ★★

```
import java.awt.geom.Point2D;

public class Optimizacion {
    /**
     * Mueve un punto en 2d en función de su velocidad, y su angulo
     *
     * @param puntoInicial
     * @param anguloGrados
     * @param espacio
     * @param tiempo
     * @return
     */
    public Point2D.Double trasladarPunto(Point2D.Double puntoInicial, float anguloGrados,
float espacio, float tiempo) {
        float velocidad = espacio / tiempo;
        double x = puntoInicial.x + (espacio / tiempo) *
Math.cos(Math.toRadians(anguloGrados));
        double y = puntoInicial.y + (espacio / tiempo) *
Math.sin(Math.toRadians(anguloGrados));
        return new Point2D.Double(x, y);
    }

    public static void main(String[] args) {
        Optimizacion ejemplo = new Optimizacion();
        Point2D.Double p = new Point2D.Double(2.0, 5.0);
        Point2D.Double nuevoPunto = ejemplo.trasladarPunto(p, 90, 5, 1);
        System.out.println(nuevoPunto);
    }
}
```

Ejercicio 5: Varios métodos a optimizar:

Optimiza los siguientes métodos.
Dificultad: ★★★

```java
import java.util.Arrays;

public class Principal {

   String getNombre() {
       String resultado = "";
       resultado = resultado + "Antonio";
       resultado = resultado + "Jose";
       resultado = resultado + "Maria";
       resultado = resultado + "Ruben";
       return resultado;
   }

   float calculo(int c) {
       float resultado = 0;
       float cto;
       do {
           cto = 5;
           resultado += c + cto;
       } while (c < 20);
       return resultado;
   }

   int[] calculo2() {
       int[] lista = { 3, 5, 7 };

       int[] l1 = lista.clone();
       Arrays.sort(l1);
       if (l1.length > 10) {
           Arrays.sort(l1);
           return l1;
       } else
```

```
            return null;
    }

    float contar(int[] listaNumeros) {
        return listaNumeros.length * 4;
    }

    float calculo3(int[] listaNumeros) {
        int valor = 0;
        for (int i = 0; i < contar(listaNumeros); i++) {
            valor += i;
        }
        return valor;
    }

}


public class Principal {

    int ejercicio(int total) {
        int contador = 0;
        int resultado;
        for (int i = 0; i < 10; i++) {
            resultado = 0;
            total += contador + i * 4;
        }
        resultado = total - 1;
        return resultado;
    }

}
```

Ejercicio 6: Pintar en pantalla números

Optimiza el siguiente método.
Dificultad: ★

```
public class Optimizacion {
    int i=1;
    void ejercioSinOptimizar() {
        while(i<100) {
            System.out.println(i);
            i++;
        }
    }
}
```

Ejercicio 7: Nombres profesores

La siguiente clase no está totalmente optimizada,
Optimízala tú. Yo al menos veo 5 optimizaciones que se pueden aplicar.
NOTA: (que no te importe si está construida con código limpio, tu solo optimízala)
Dificultad: ★ ★ ★

```
public class Optimiza {
   String nombres[] = { "Angel", "Bea", "Pepe" };

   /**
    * Este método comprueba si un nombre está en la lista
    *
    * @param nombre valor a comprobar
    * @return true si el nombre está en la lista
    */
   boolean compruebaSiEsta(String nombreBuscado) {
       boolean esta = false;
       for (String nombre : nombres) {
           if (nombreBuscado.equals(nombre)) {
               esta = true;
           }
       }
       return esta;
   }

   /**
    * Este método hace un calculo sobre la lista
    *
    * @return un valor cualquiera
    */
   float calculo() {
       int suma = 0;
       int valor = 8;
       int cto = 9;
       for (String nombre : nombres) {
           suma += nombre.length();
           valor += suma;
           cto = 5;
       }
       return suma / 8 + cto;
   }

   /**
    * Este método devuelve todos los nombres de la lista concatenados uno a
    * continuación del otro
    *
```

```
     * @return los nombres concatenados
     */
    String obtenerNombresConcatenados() {
        String resultado = "";
        for (String nombre : nombres) {
            resultado += nombre;
        }
        return resultado;
    }

}
```

Ejercicio 8: Operaciones con arrays de nombres

La siguiente clase no está totalmente optimizada,
Optimiza el siguiente código.
También refactorízalo en la medida de lo posible.
Dificultad: ★★★

```
import java.io.Reader;
import java.util.Arrays;
import java.util.Scanner;

public class Nombres {
    String[] lista = { "Angel", "Pepe", "Bea" };

    void Insert(String nuevoNombre) {
        String[] temp = Arrays.copyOf(lista, lista.length + 1);
        temp[lista.length] = nuevoNombre;
        lista = temp;
    }

    void InsertMultiple(String nombre1, String nombre2, String nombre3, String
nombre4, String nombre5,
            String nombre6) {
        if (nombre1 != null)
            Insert(nombre1);
        if (nombre2 != null)
            Insert(nombre2);
        if (nombre3 != null)
```

```
            Insert(nombre3);
        if (nombre4 != null)
            Insert(nombre4);
        if (nombre5 != null)
            Insert(nombre5);
        if (nombre6 != null)
            Insert(nombre6);
    }

    void removeLastElement() {
        if (lista.length > 0) {
            String[] temp = Arrays.copyOf(lista, lista.length - 1);
            lista = temp;
        } else {
            System.out.println("No hay elementos para borrar");
        }
    }

    void clear() {
        lista = new String[] {};
    }

    void sortNames() {
        String m = "";
        for (int i = 0; i < lista.length; i++) {
            m += lista[i];
        }
        Arrays.sort(lista);
    }

    void printAllNames() {
        System.out.println("---- Lista de nombres -----");
        for (String nombre : lista) {
            System.out.println(nombre);
        }
    }

    void rellenarNombres() {
        Scanner in = new Scanner(System.in);
        System.out.println("Introduce la cantidad de nombres a leer: ");
        int cantidad = in.nextInt();
        for (int i = 0; i < cantidad; i++) {
```

```
            System.out.println("Introduce el nombre: ");
            String nombre = in.next();
            Insert(nombre);
        }
        in.close();
    }

    public static void main(String[] args) {
        Nombres n = new Nombres();
        n.rellenarNombres();
        n.printAllNames();
        n.Insert("Miguel");
        n.printAllNames();
        n.removeLastElement();
        n.printAllNames();
        n.clear();
        n.printAllNames();
        n.removeLastElement();
        n.printAllNames();
        n.Insert("Paula");
        n.Insert("Carmen");
        n.Insert("Ana");
        n.sortNames();
        n.printAllNames();

    }
}
```

Ejercicio 9: Cálculos de frases

La siguiente clase no está totalmente optimizada,
Optimiza el siguiente código.
También refactorízalo en la medida de lo posible.

Se parte del siguiente programa que almacena un conjunto de frases y realiza una serie de cálculos sobre ellas.

Las frases no se van a modificar (ni se añaden o quitan frases), con lo cual **no será necesario** usar un ArrayList, con un array tradicional (como el que se usa en el programa) es suficiente.
Puede que no todos los métodos requieren ser optimizados.
Pon encima de cada método, una breve explicación de lo que has hecho para optimizar.
Dificultad: ★★★

```
import java.util.Arrays;

public class Optimiza {
   String frases[] = { "Hola", "Angel", "que tal estas", "Angel", "Pepe", "que tal
estoy"};

   /* Este método busca si una frase está en la lista de frases */
   boolean busca(String fraseBuscar) {
       boolean siEsta = false;
       boolean noEsta = true;
       for (String frase : frases) {
           if (frase.equals(fraseBuscar)) {
               siEsta = true;
               noEsta = false;
           }
       }
       noEsta = !siEsta;
       return siEsta;
   }

   /* Devuelve la frase mas larga (si hay varias de igual tamaño devuelve la
primera que se encuentra) */
   String masLarga() {
       String masLarga = frases[0];
       for(int i=0;i<frases.length;i++){
           if (frases[i].length() > masLarga.length()) {
               masLarga = frases[i];
           }
       }
       return masLarga;
   }
   /* Este cuenta cuantos frases de como minimo X letras */
   int cuentaFrasesMinimoLetras(int minimoLetras) {
       int contador = 0;
       int num = contador;
       for (String frase : frases) {
           if (frase.length() >= minimoLetras) {
               num = contador;
```

```
                contador++;
            }
        }
        num = num / frases.length;
        return contador;
    }


     /* Calcula la media de caracteres de todas las frases*/
    // En el ejemplo la media de todos los caracteres es 4+5+13+5+4+13  /  6  = 7.33
     float mediaCaracteres(){
        float suma = 0;
        for(String frase: frases){
            suma+=frase.length();
        }
        return suma / frases.length;
    }

    /* Por cada frase, devuelve el numero de caracteres de cada frase dividido por
la media de caracteres totales  */
    // Contamos numero de caracteres de cada frase = [4, 5, 13, 5, 4, 13]
    // Numero total de frases = 6
    // Media total de caracteres  = 4+5+13+5+4+13  /  6  = 7.33
    // Dividimos el numero de caracteres de cada frase por la media total de
caracteres
    // [0.5714286, 0.71428573, 1.8571428, 0.71428573, 0.5714286, 1.8571428]
    float[] numerosCaracteresEntreMediaTotal(){
        float[] resultado = new float[frases.length];
        for(int i=0;i<frases.length;i++){
            resultado[i] = frases[i].length() / mediaCaracteres();
        }
        return resultado;
    }

    /* Devuelve la frase que mas se repite (moda) */
    String moda() {
        int maximaVecesQueSeRepite = 0;
        String moda = "";
        for (int i = 0; i < frases.length; i++) {
            int vecesQueSeRepite = 0;
            for (int j = 0; j < frases.length; j++) {
                if (frases[i] == frases[j])
                    vecesQueSeRepite++;
            }
            if (vecesQueSeRepite > maximaVecesQueSeRepite) {
```

```
                moda = frases[i];
                maximaVecesQueSeRepite = vecesQueSeRepite;
            }
        }
        return moda;
    }

    int calcula(){
        int moda1 = moda().length();
        int moda2 = moda().length();
        int l = 0;
        for(int i=0;i<frases.length;i++){
            int mediaEntera = (int)mediaCaracteres();
            l += frases[i].length() + Math.abs(moda1) + Math.abs(moda2) + mediaEntera;
        }
        return l/8;
    }


    public static void main(String[] args) {
        new Optimiza();
    }

    public Optimiza() {
        System.out.println("Frases: " + Arrays.toString(frases));
        System.out.println("Contiene la frase Angel:" + busca("Angel"));
        System.out.println("Contiene la frase Bea:" + busca("Bea"));
        System.out.println("Hay " + cuentaFrasesMinimoLetras(5) + " frases de mínimo
5 letras");
        System.out.println("La frase mas larga es: " + masLarga());
        System.out.println("La moda vale:" + moda());
    }
}
```